

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-180025

(43) 公開日 平成8年(1996)7月12日

(51) Int.Cl. ⁴	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/16	3 4 0 A			
9/46	3 6 0 C			

審査請求 未請求 請求項の数5 O L (全 13 頁)

(21) 出願番号 特願平6-318105

(22) 出願日 平成6年(1994)12月21日

(71) 出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(72) 発明者 岐津 俊樹

東京都青梅市末広町2丁目9番地 株式会

社東芝青梅工場内

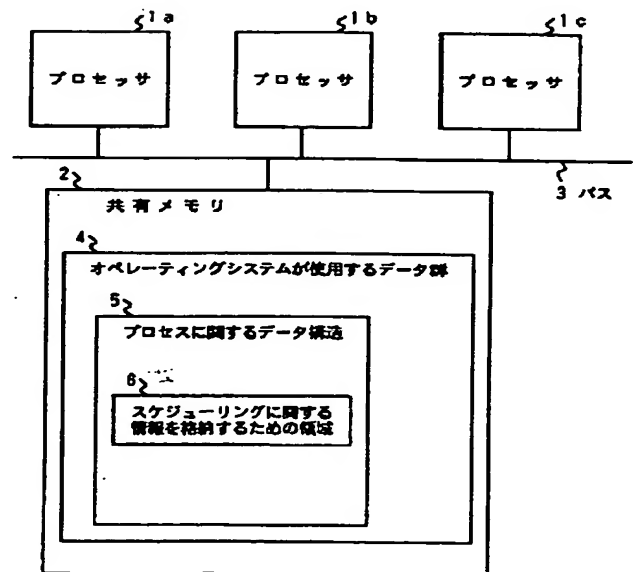
(74) 代理人 弁理士 鈴江 武彦

(54) 【発明の名称】 スケジューリング装置

(57) 【要約】

【目的】 アドレス空間を共有するプロセスが同一のプロセッサで実行されるようにスケジューリングするスケジューリング装置を提供する。

【構成】 複数のプロセッサ 1 a ~ 1 c がメモリ 2 を共有するマルチプロセッサシステムであって、アドレス空間を共有する複数のスレッドから構成されるプロセスを実行するマルチプロセッサシステムのスケジューリング装置において、上記スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位にデータ構造 5 の領域 6 に格納する手段と、上記プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、この領域 6 に格納された情報からそのスレッドが当該プロセッサにて実行可能であるか否かを判断する手段とを具備し、同一のプロセスに属するスレッドすべてを同一のプロセッサで実行させることを特徴とする。



1

【特許請求の範囲】

【請求項 1】 複数のプロセッサがメモリを共有するマルチプロセッサシステムであって、アドレス空間を共有する複数のスレッドから構成されるプロセスを実行するマルチプロセッサシステムのスケジューリング装置において、

上記スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位に管理する手段と、上記プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、上記格納された情報からそのスレッドが当該プロセッサにて実行可能であるか否かを判断する手段とを具備し、同一のプロセスに属するスレッドすべてを同一のプロセッサで実行させることを特徴とするマルチプロセッサシステムのスケジューリング装置。

【請求項 2】 上記管理手段は、複数のプロセス間でアドレス空間の共有が発生した際に、このプロセス群単位に上記スレッドそれぞれが実行されるべきプロセッサを示す情報を管理する手段を具備し、同一のプロセス群に属するスレッドすべてを同一のプロセッサで実行させることを特徴とする請求項 1 記載のマルチプロセッサシステムのスケジューリング装置。

【請求項 3】 上記管理手段は、複数のプロセス群間でアドレス空間の共有が発生した際に、アドレス空間の共有を要求した側のプロセスが形成するすべてのプロセス群の情報をアドレス空間の共有を要求された側のプロセスが形成するプロセス群の情報を更新する手段を具備し、アドレス空間を共有するプロセス群に属するスレッドすべてを同一のプロセッサで実行させることを特徴とする請求項 2 記載のマルチプロセッサシステムのスケジューリング装置。

【請求項 4】 上記管理手段は、いずれかのプロセスがアドレス空間の共有を解消した際に、このプロセスが形成するプロセス群同士のアドレス空間の共有が解消されるか否かを判定し、プロセス群同士のアドレス空間の共有が解消される場合に、これらのプロセス群各々に個別の情報を割り振る手段を具備してなることを特徴とする請求項 3 記載のマルチプロセッサシステムのスケジューリング装置。

【請求項 5】 上記管理される情報は、上記複数のプロセッサのいずれのプロセッサで実行してもかまわない旨を示す情報を含むことを特徴とする請求項 1、2、3 又は 4 記載のマルチプロセッサシステムのスケジューリング装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、複数のプロセッサがメモリを共有するマルチプロセッサシステムに適用して好適なスケジューリング装置に係り、特にキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることによりシステム性能の向上を図るこ

2

とを可能とするスケジューリング装置に関する。

【0002】

【従来の技術】 従来の複数のプロセッサがメモリを共有するマルチプロセッサシステムにおいては、共有メモリを使用することを前提にオペレーティングシステムが作成されており、通常は、処理性能を向上させるために、メモリより高速にアクセス可能なキャッシュメモリを設け、メモリにアクセスした結果をこのキャッシュメモリに蓄積するようにしている。

【0003】 しかし、プロセススケジューラは、複数のプロセス間でアドレス空間を共有している場合でも、スケジュールの際にそれを考慮することがなく、そのためにアドレス空間を共有しているようなプロセス同士が別々のプロセッサで実行されることがあり、このときに共有しているアドレス空間に書き込み等の更新が行われると、キャッシュメモリの一貫性を保持するための排他制御を行う必要が発生し、これがオーバーヘッドとなりシステム全体の効率を低下させていた。

【0004】

【発明が解決しようとする課題】 上述したように、従来の複数のプロセッサがメモリを共有するマルチプロセッサシステムのスケジューリング装置においては、プロセスをスケジューリングする際に、プロセス間のアドレス空間の共有を考慮に入れることがなかったため、キャッシュメモリ及び共有メモリの一貫性を保持するためのオーバーヘッドを発生させるといった問題があった。

【0005】 本発明は上記実情に鑑みなされたものであり、プロセス間でアドレス空間を共有している場合に、それらのプロセスを同一のプロセッサにスケジューリングして実行することにより、キャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させ、システム性能の向上を図ることを可能とするスケジューリング装置を提供することを目的とする。

【0006】

【課題を解決するための手段】 本発明は、複数のプロセッサがメモリを共有するマルチプロセッサシステムであって、アドレス空間を共有する複数のスレッドから構成されるプロセスを実行するマルチプロセッサシステムのスケジューリング装置において、上記スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位に管理する手段と、上記プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、上記格納された情報からそのスレッドが当該プロセッサにて実行可能であるか否かを判断する手段とを具備し、同一のプロセスに属するスレッドすべてを同一のプロセッサで実行させることを特徴とする。

【0007】 また、本発明は、上記管理手段が、複数のプロセス間でアドレス空間の共有が発生した際に、このプロセス群単位に上記スレッドそれぞれが実行されるべきプロセッサを示す情報を管理する手段を具備し、同一

3

のプロセス群に属するスレッドすべてを同一のプロセッサで実行させることをすることを特徴とする。

【0008】また、本発明は、上記管理手段が、複数のプロセス群間でアドレス空間の共有が発生した際に、アドレス空間の共有を要求した側のプロセスが形成するすべてのプロセス群の情報をアドレス空間の共有を要求された側のプロセスが形成するプロセス群の情報を更新する手段を具備し、アドレス空間を共有するプロセス群に属するスレッドすべてを同一のプロセッサで実行させることを特徴とする。

【0009】また、本発明は、上記管理手段が、いずれかのプロセスがアドレス空間の共有を解消した際に、このプロセスが形成するプロセス群同士のアドレス空間の共有が解消されるか否かを判定し、プロセス群同士のアドレス空間の共有が解消される場合に、これらのプロセス群各々に個別の情報を割り振る手段を具備してなることを特徴とする。また、本発明は、上記管理される情報が、上記複数のプロセッサのいずれのプロセッサで実行してもかまわない旨を示す情報を含むことを特徴とする。

【0010】

【作用】本発明の構成によれば、管理手段が、スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位に管理しており、プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、この管理された情報を参照して自身が実行可能なスレッドであるか否かを判定する。この判定により実行可能な具体例としては、

(1) その情報がいずれのプロセッサで実行してもかまわない旨を示している場合。

(2) その情報が自身のプロセッサを示している場合。

【0011】この情報はプロセス単位に管理されるため、同一のプロセスに属するスレッドはすべて同一のプロセッサで実行されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【0012】また、本発明の構成によれば、複数のプロセス間でアドレス空間の共有が発生した際に、上述した管理手段が、アドレス空間を共有するこれら複数のプロセスにより形成されるプロセス群単位にスレッドそれぞれが実行されるべきプロセッサを示す情報を管理する。

【0013】例えば、Aというプロセスが使用するアドレス空間について、B及びCというプロセスが共有を要求した場合、このA、B及びCのプロセスを一つのプロセス群として、このプロセス群単位に情報を管理する。

【0014】これにより、アドレス空間を共有するA、B及びCのプロセスで形成されるプロセス群に属するスレッドはすべて同一のプロセッサで実行されることになり、これらのプロセスがアドレス空間を共有することに

4

より発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【0015】また、本発明の構成によれば、複数のプロセス群間でアドレス空間の共有が発生した際に、上述した管理手段が、アドレス空間の共有を要求した側のプロセスが形成するすべてのプロセス群の情報を、アドレス空間の共有を要求された側のプロセスが形成するプロセス群の情報を更新する。

【0016】例えば、いま、A、B及びCにより構成される第1のプロセス群と、X、Y及びZにより構成される第2のプロセス群が存在し、この第1及び第2のプロセス群は、それぞれ個別のプロセッサで実行されるものとする。そして、第2のプロセス群を形成するXのプロセスが、第1のプロセス群を形成するAのプロセスの使用アドレス空間の共有を要求した場合を考える。

【0017】このとき、管理手段は、Aのプロセスが形成する第1のプロセス群の情報と、Xのプロセスが形成する第2のプロセス群の情報とが等しいか否かを判断し、異なっていれば、第2のプロセス群の情報を第1のプロセス群の情報に更新する。

【0018】この結果、A、B、C、X、Y及びZのプロセスはすべて同一のプロセッサで実行されることになり、A及びXのプロセスがアドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【0019】なお、Xのプロセスが、Y及びZのプロセスと形成する第2のプロセス群以外に他のプロセスとプロセス群を形成していた場合には、そのすべてのプロセス群の情報を、第1のプロセス群の情報に更新することにより、これらのオーバーヘッドを減少させる。

【0020】また、本発明の構成によれば、いずれかのプロセスがアドレス空間の共有を解消した際に、このプロセスが形成するプロセス群同士のアドレス空間の共有が解消されるか否かを判定し、プロセス群同士のアドレス空間の共有が解消される場合に、これらのプロセス群各々に個別の情報を割り振る。

【0021】ここでは、上述したようなA、B及びCにより形成される第1のプロセス群と、X、Y及びZにより形成される第2のプロセス群がアドレス空間を共有している場合、具体的には、A及びXのプロセスがアドレス空間を共有している場合を前提として、このA及びXのプロセスによるアドレス空間の共有が解消された場合を考える。

【0022】このとき、管理手段は、この第1及び第2のプロセス群同士のアドレス空間の共有が解消されるか否かを判定する。この判定でプロセス群同士のアドレス空間の共有の解消が拒否される例としては、さらに、B及びYのプロセスがアドレス空間を共有している場合等

が挙げられる。

【0023】そして、プロセス群同士のアドレス空間の共有が解消される場合には、もはや同じプロセッサで実行すべき理由がないため、第2のプロセス群の情報を他の情報に割り振ることが可能となる。この割り振りは、プロセッサの稼働率に基づく等、所定の規則に従って行えばよい。これにより、システム全体の効率をより向上させることが可能となる。

【0024】

【実施例】以下図面を参照して本発明の実施例を説明する。まず、図1を参照して本発明の第1実施例を説明する。図1は第1実施例に係るスケジューリング装置の概略構成を示す図である。

【0025】図1に示すように、同実施例に係るスケジューリング装置は、プロセッサ1a~1cと共有メモリ2とがバス3を介して接続されている。また、共有メモリ2には、オペレーティングシステムが実行する際に必要となるデータ群4が格納されている。そして、このデータ群4には、プロセスに関するデータ構造5が含まれ、さらに、このプロセスに関するデータ構造5には、スケジューリングに関する情報を格納するための領域6が含まれている。このスケジューリングに関する情報を格納するための領域6には、各プロセスに属するスレッドが実行されるべきプロセッサのIDが格納される。また、プロセッサ1a~1cのいずれのプロセッサで実行されてもかまわない場合には、その旨を示す値が格納される。

【0026】ここで、あるプロセッサが、実行キューに投入されたスレッドを処理しようとしたとき、そのプロセッサは、この領域6に格納された情報を参照して、自身が実行可能であるか否かを判定する。この場合、領域6に格納された情報が、

(1) いずれのプロセッサで実行してもかまわない旨を示している。

(2) その情報が自身のプロセッサを示している。
ときにそのスレッドを実行する。

【0027】これにより、スレッドを実行すべきプロセッサの情報がプロセス単位に管理されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【0028】次に、図2乃至図4を参照して本発明の第2実施例を説明する。図2に示すように、同実施例に係るスケジューリング装置は、プロセッサ1a~1cと共有メモリ2とがバス3を介して接続されている。また、共有メモリ2には、オペレーティングシステムが実行する際に必要となるデータ群4が格納されている。そして、このデータ群4には、プロセスに関するデータ構造5及び複数のプロセス間におけるアドレス空間の共有に関するデータ構造8が含まれ、さらに、このプロセスに

関するデータ構造5には、スケジューリングに関する情報を格納するための領域6とアドレス空間の共有に関するデータ構造8を指し示すポインタ7とが含まれている。

【0029】図3及び図4は、このアドレス空間の共有に関するデータ構造8を具体的に示した概念図である。ここでは、図3に示すように、システム中にプロセス9a、9b及び9cがあり、それらがあるアドレス空間を共有する場合を考える。図4に示すように、これらのプロセスに対応するデータ構造は、それぞれ13a~13cである。そして、このプロセス群に対しprocgroupなるデータ構造11を割り当てる。また、このプロセス群に属する各プロセスに対応してpgsegなるデータ構造12a~12cを割り当てる。そして、これらはリンクされた形で保持される。

【0030】このprocgroup 11は、pgseg データ構造を指し示すためのポインタであるpg-pgseglink、これを更新する際に排他制御を行うためのロックであるpg-pgseglock、及びこのprocgroup 11に属するプロセスが実行されるべきプロセッサのIDが格納されるpg-cpuなるメンバにより構成される。一方、pgseg 12a~12cには、次のpgseg 構造を指し示すためのポインタであるpgseg-link、procgroup 11を指し示すためのポインタであるpgseg-pg、及びこのpgseg が属するプロセスのデータ構造13a~13cを指し示すためのポインタであるpgseg-procpなるメンバで構成される。また、各プロセスのデータ構造13a~13cには、対応するpgseg データ12a~12cを指し示すためのポインタであるp-pgsegpなるメンバが存在する。このp-pgsegpは、アドレス空間を共有していない場合には初期値が格納されている。

【0031】ここで、図5を参照してプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明する。図5はプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャートである。

【0032】まず、オペレーティングシステムは、procgroup データ構造11及びpgseg データ構造12a~12cを割り当て（図5のステップA1~A2）、それぞれのデータ構造を適切にリンクすることにより、各メンバの初期化を行う（図5のステップA3）。最後に、オペレーティングシステムは、これらのプロセス群をどのプロセッサで実行すべきかを決定し、procgroup データ構造11のメンバであるpg-cpuにそのプロセッサIDを格納する（図5のステップA4）。

【0033】次に、図6を参照してオペレーティングシステムがスレッドを実行する際の動作手順を説明する。図6はオペレーティングシステムがスレッドを実行する際の動作手順を説明するためのフローチャートである。

【0034】実行キューに投入されたスレッドが存在す

7

る場合（図6のステップB1のY）、オペレーティングシステムは、そのスレッドが属するプロセスのデータ構造13a~13cを調べ（図6のステップB2）、メンバp-pgsegpが初期値でない場合には（図6のステップB2のN）、対応するpgseg データ構造12a~12cのpgseg-pgからprocgroup データ構造11を求め（図6のステップB3）、そのメンバであるpg-cpuを参照して予め設定されたプロセッサIDを認識する（図6のステップB4）。

【0035】ここで、オペレーティングシステムは自身のIDとそのプロセッサIDとが一致しているか否かを判断し（図6のステップB5）、一致していればそのスレッドを実行し（図6のステップB5のY）、不一致であれば、次の実行待ちスレッドについて同様の処理を繰り返す（図6のステップB5のN）。

【0036】これにより、図3に示すプロセス9a、9b及び9cは、同一のプロセッサで実行されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【0037】次に、図7乃至図9を参照して本発明の第3実施例を説明する。図7及び図8は、同実施例におけるアドレス空間の共有に関するデータ構造8を具体的に示した概念図である。

【0038】ここでは、図7に示すように、システム中にプロセス9a、9b、9c、9d及び9eがあり、それらがあるアドレス空間を共有する場合、具体的には、プロセス9a、9b、9c及び9dがアドレス空間10xを、プロセス9d及び9eがアドレス空間10yを、プロセス9a及び9bがアドレス空間10zをそれぞれ共有する場合を考える。

【0039】図8に示すように、これらのプロセスに対応するデータ構造は、それぞれ13a~13eである。そして、このプロセス群それぞれに対しprocgroup なるデータ構造11x~11zを割り当てる。また、このプロセス群に属する各プロセスに対応してpgseg なるデータ構造12a1~12e1を割り当てる。そして、これらはリンクされた形で保持される。

【0040】このprocgroup 11x~11zは、pgseg データ構造を指し示すためのポインタであるpg-pgseglink、これを更新する際に排他制御を行うためのロックであるpg-pgseglock、及びこのprocgroup 11x~11zに属するプロセスが実行されるべきプロセッサのIDが格納されるpg-cpuなるメンバにより構成される。一方、pgseg 12a1~12e1には、次のpgseg 構造を指し示すためのポインタであるpgseg-link、同一プロセス内で、別のアドレス空間を共有している場合にそのpgseg データ構造を指し示すためのポインタであるpgseg-plink、procgroup 11x~11zを指し示すためのポインタであるpgseg-pg、及びこのpgseg が属するプロセスの

8

データ構造13a~13eを指し示すためのポインタであるpgseg-procp なるメンバで構成される。また、各プロセスのデータ構造13a~13eには、対応するpgseg データ12a1~12e1を指し示すためのポインタであるp-pgsegpなるメンバが存在する。このp-pgsegpは、アドレス空間を共有していない場合には初期値が格納されている。また、これらすべてのデータ構造を更新する際に排他制御が必要となるが、そのためのロックを示すものがpglock14である。

【0041】ここで、図9を参照してプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明する。図9はプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャートである。

【0042】初めてアドレス空間の共有が発生した場合には、図5で示した手順により処理を行う。そして、ここでは、既に共有されているアドレス空間に対して新たに共有を要求する場合を説明する。

【0043】この場合、まず、その新たにアドレス空間の共有を要求したプロセスに対応するpgseg データ構造12a1~12e1を割り当てる（図9のステップC1）。そして、この割り当てたpgseg データ構造と、アドレス空間に対応したprocgroup データ構造11x~11zとの間をリンクさせ、メンバの初期化を行う（図9のステップC2）。

【0044】ここで、互いに別のアドレス空間を共有しているプロセス同士が新たにアドレス空間を共有する場合には（図9のステップC3のY）、双方のデータ構造を合わせ、グルーピングする必要がある。この際、アドレス空間の共有を要求した側のprocgroup データ構造の後リンク（pg-forw）又は前リンク（pg-back）をたどり、共有する相手側プロセスの属するprocgroup データ構造が出現してこないかを確認する（図9のステップC4~C6）。もし、相手側プロセスの属するprocgroup データ構造が出現した場合には（図9のステップC6のY）、すでに共有によるデータ構造のグルーピングが行われていることになるので、新たに処理を行う必要はない。一方、相手側プロセスの属するprocgroup データ構造が出現しなかった場合には、グルーピングを行う（図9のステップC7）。この場合、要求側プロセスの属するprocgroup データ構造と相手側プロセスの属するprocgroup データ構造とをリンクさせ、要求側プロセスの属するprocgroup データ構造すべてのpg-cpuメンバを相手の属するprocgroup データ構造のpg-cpuメンバに更新する。

【0045】これにより、図7に示すプロセス9a、9b、9c、9d及び9eは、同一のプロセッサで実行されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【0046】次に、図10を参照して上述したプロセス間のアドレス空間の共有が解消された場合のオペレーティングシステムの動作手順を説明する。図10はプロセス間のアドレス空間の共有が解消された場合のオペレーティングシステムの動作手順を説明するためのフローチャートである。

【0047】ここでは、図7の状態から、プロセス9dが共有アドレス10xの共有を解消する場合を考える。この場合、まずオペレーティングシステムは、共有の解消するアドレス空間に対応するprocgroup データ構造11xから、pgseg データ構造12d1を取り除く(図10のステップD1)。この際、そのpgseg データ構造が他のprocgroupデータ構造にも存在する場合には、このpgseg データ構造同士のリンクも適切に更新する。この例では、pgseg データ構造12d2が存在するため、このpgsegデータ構造12d2のpgseg-plink を初期化しておく。

【0048】次に、procgroup データ構造のリンクをたどり、先頭のprocgroup を検索する(図10のステップD2)。この例では11xが該当する。そして、ここに繋がれているすべてのpgseg に対して、各々のpgseg-plink をたどり、共有関係のあるprocgroup データ構造を求め、これをprocgroup データ構造のリンクpg-linkに加える。この例では、procgroup データ構造11xからpg-pgseglinkをたどり、pgseg 12a1がまず検索される。そして、このpgseg 12a1は、pgseg 12a2に対するリンクをもっているので、pgseg 12a2の属するprocgroupデータ構造であるprocgroup データ構造11zをprocgroup データ構造11xのリンクpg-linkに加える(図10のステップD4)。オペレーティングシステムは、この動作を繰り返し、pgseg 12b1及びpgseg 12c1についての処理が終了したら、procgroup データ構造11xのpg-link をたどり、次のprocgroup データ構造について同様の処理を繰り返す。

【0049】この処理がすべて終了した時点で、プロセス群を分割できる場合には、pg-link でつながれているprocgroup 群とつながれていないprocgroup 群とに分かれるので(図10のステップD6のY)、つながれていないprocgroup 群をpg-forw及びpg-back でリンクする

(図10のステップD7)。また、つながれているprocgroup 群に対しても、pg-forw 及びpg-back を設定しておくとともに、pg-link を初期化しておく。

【0050】次に、分割されたグループの一方を、適切なプロセッサで実行させるため、pg-cpuを更新する。なお、pg-link でつながれていないprocgroup 群が存在しなかった場合には、分割ができない場合ということにな

り、この場合には、pg-form とpg-back を設定してpg-link を初期化する。これにより、システム全体の効率をより向上させることが可能となる。

【0051】

【発明の効果】以上詳記したように本発明によれば、プロセス間でアドレス空間を共有している場合に、それらのプロセスを同一のプロセッサにスケジューリングして実行することにより、キャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させ、システム性能の向上を図ることが可能となる。

【0052】また、同一プロセッサで実行すべき必要性がなくなった場合に、実行プロセッサを適切に割り振るため、システム全体の効率をより向上させることが可能となる。

【図面の簡単な説明】

【図1】本発明の第1実施例に係るスケジューリング装置の概略構成を示す図。

【図2】本発明の第2実施例に係るスケジューリング装置の概略構成を示す図。

【図3】第2実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

【図4】第2実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

【図5】プロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャート。

【図6】オペレーティングシステムがスレッドを実行する際の動作手順を説明するためのフローチャート。

【図7】第3実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

【図8】第3実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

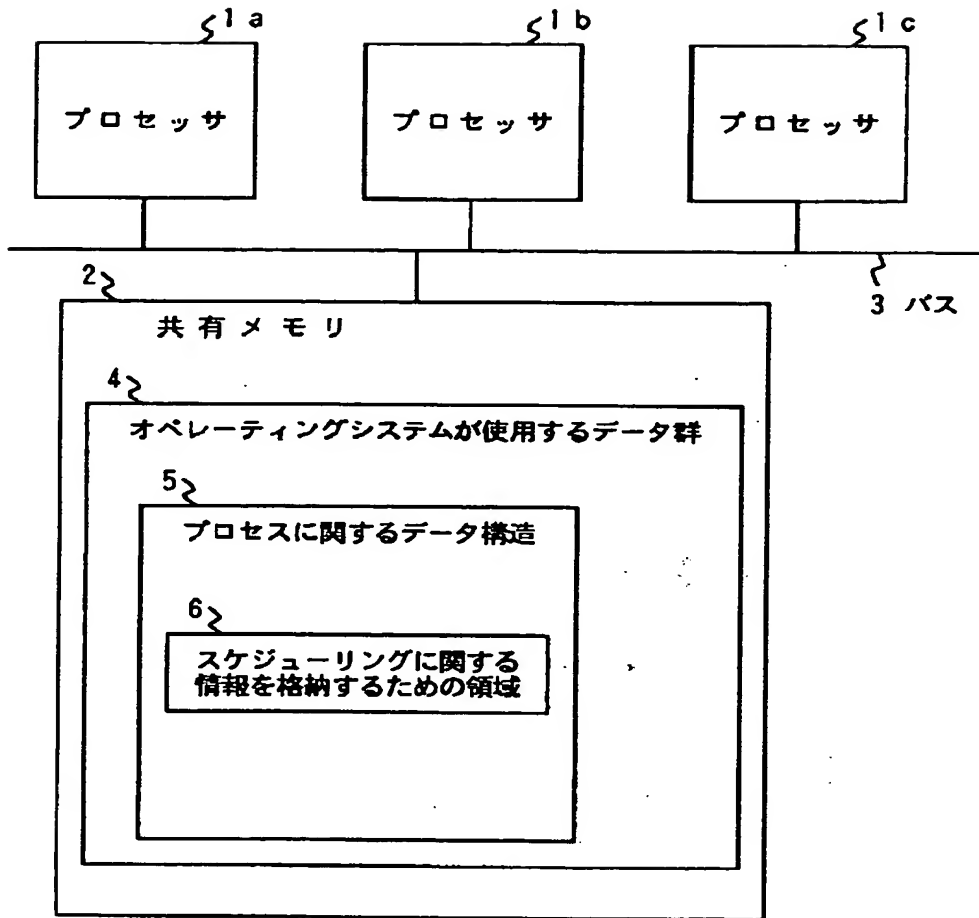
【図9】プロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャート。

【図10】プロセス間のアドレス空間の共有が解消された場合のオペレーティングシステムの動作手順を説明するためのフローチャート。

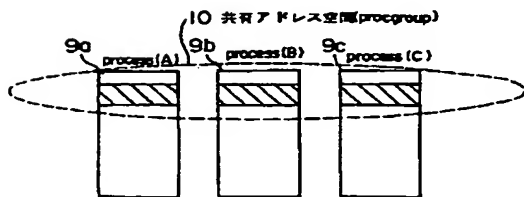
【符号の説明】

1a~1c…プロセッサ、2…共有メモリ、3…バス、4…オペレーティングシステムが使用するデータ群、5…プロセスに関するデータ構造、6…スケジュールに関する情報を格納するための領域、7…アドレス空間の共有に関するデータ構造へのポインタ、8…アドレス空間の共有に関するデータ構造。

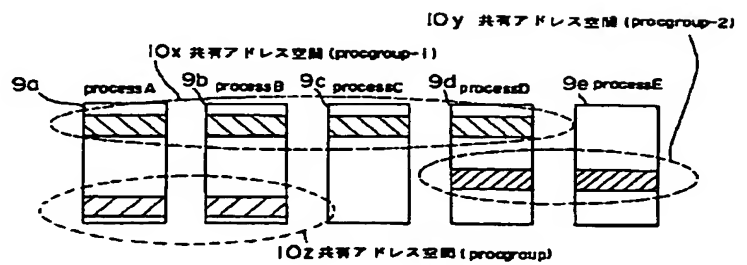
【図 1】



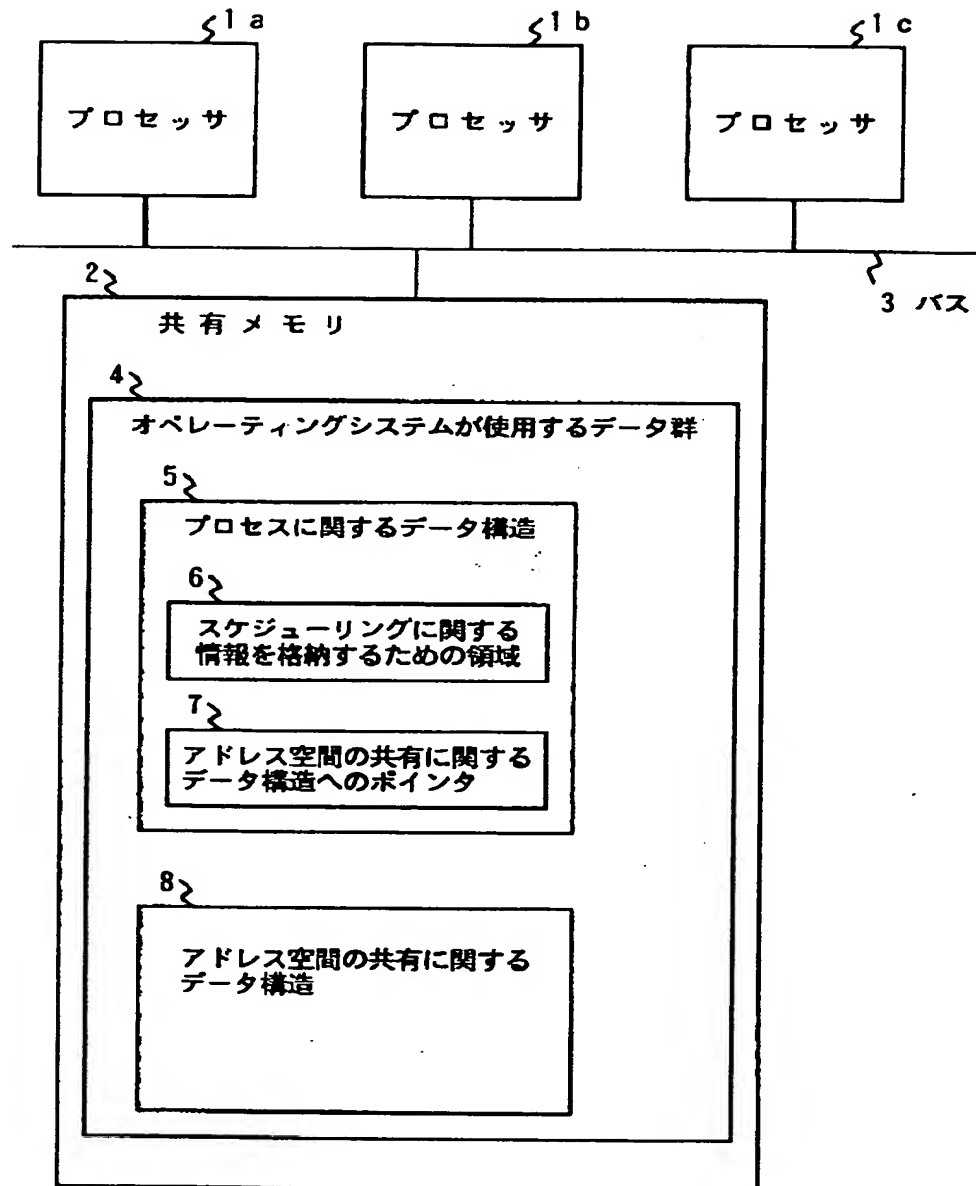
【図 3】



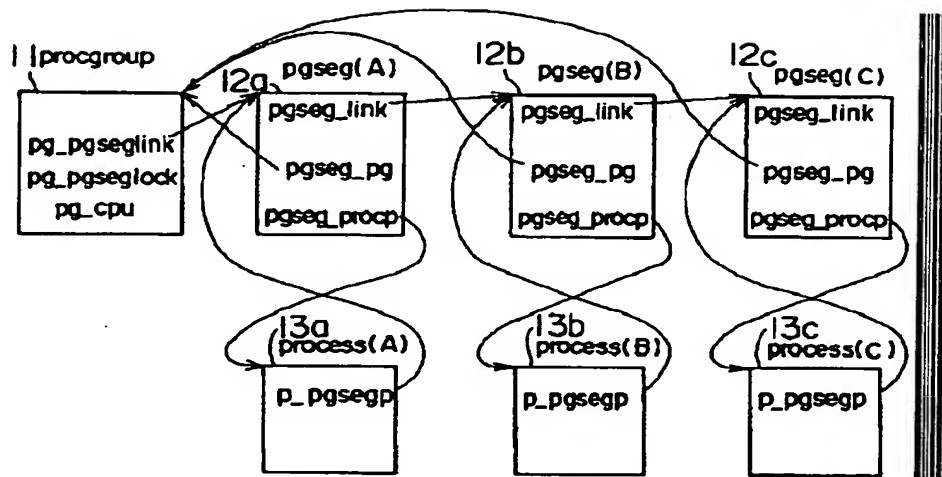
【図 7】



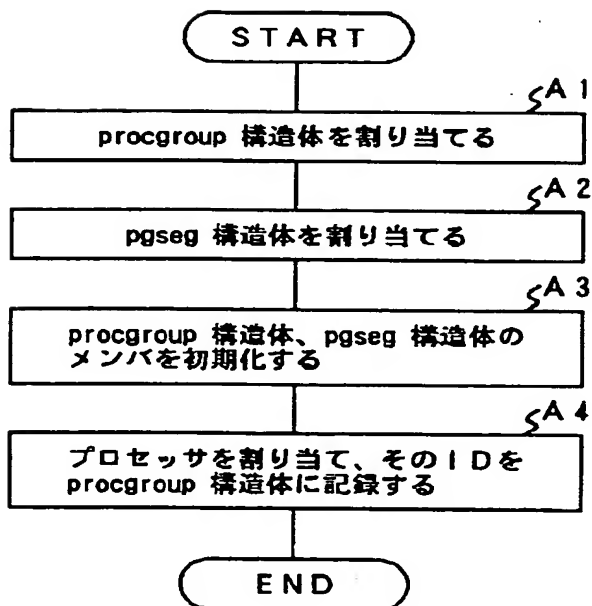
【図 2】



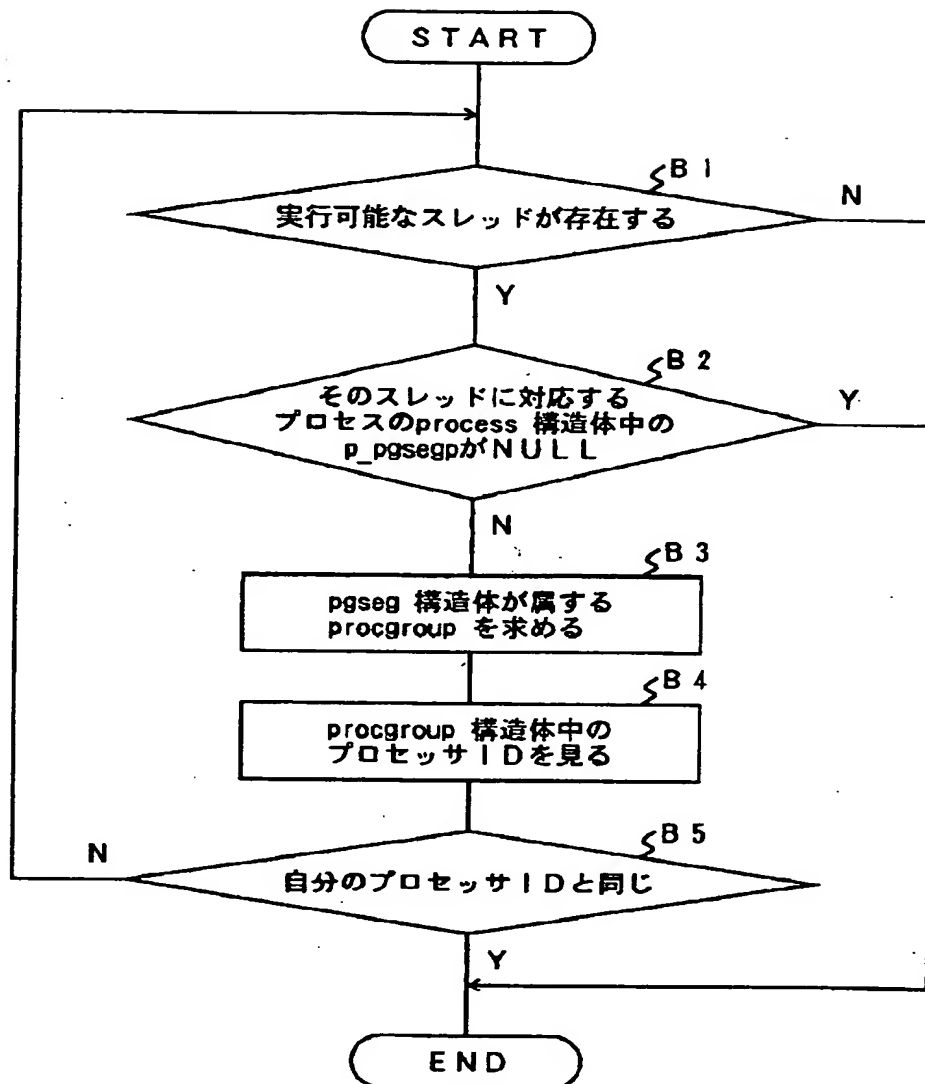
【図 4】



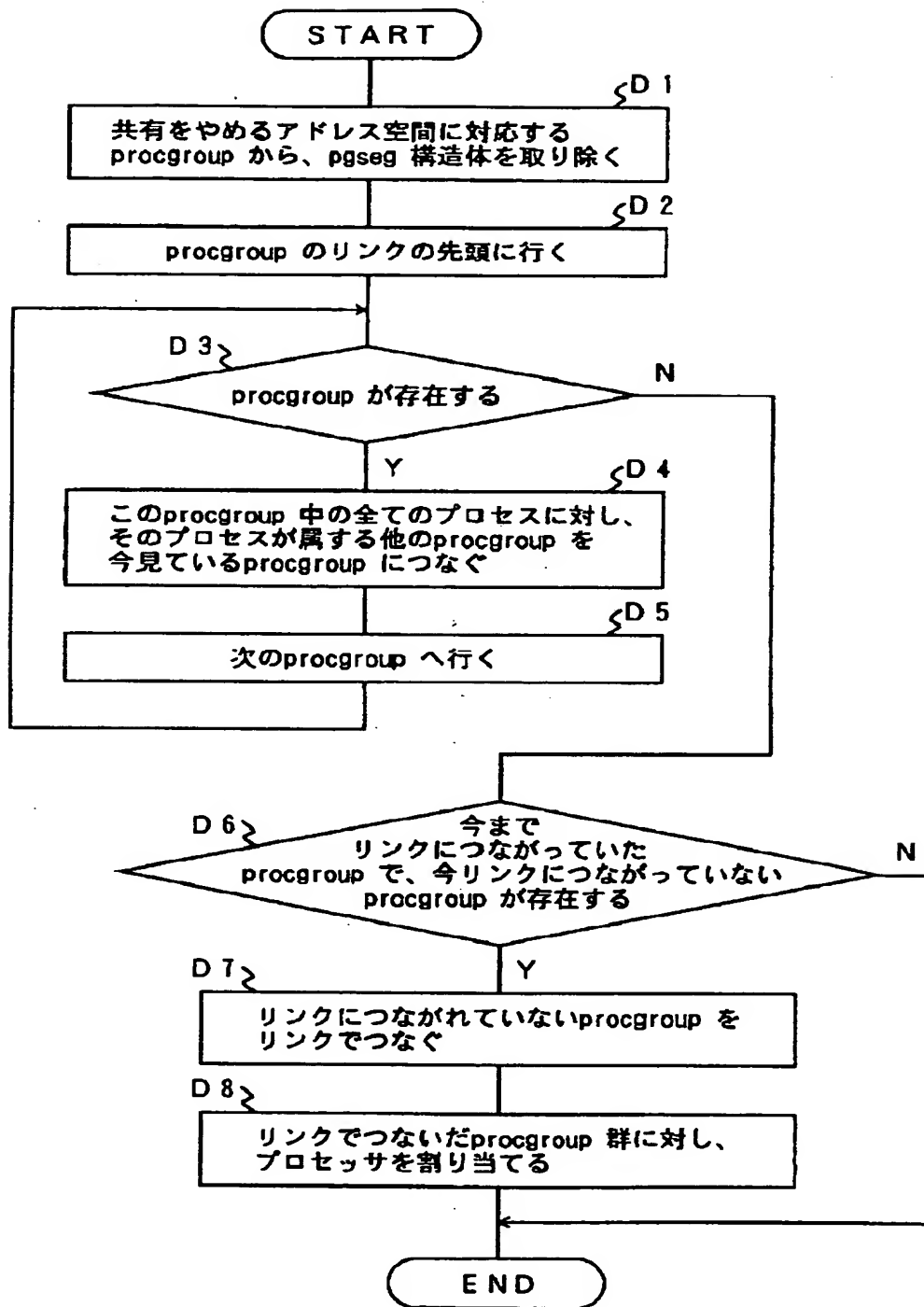
【図 5】



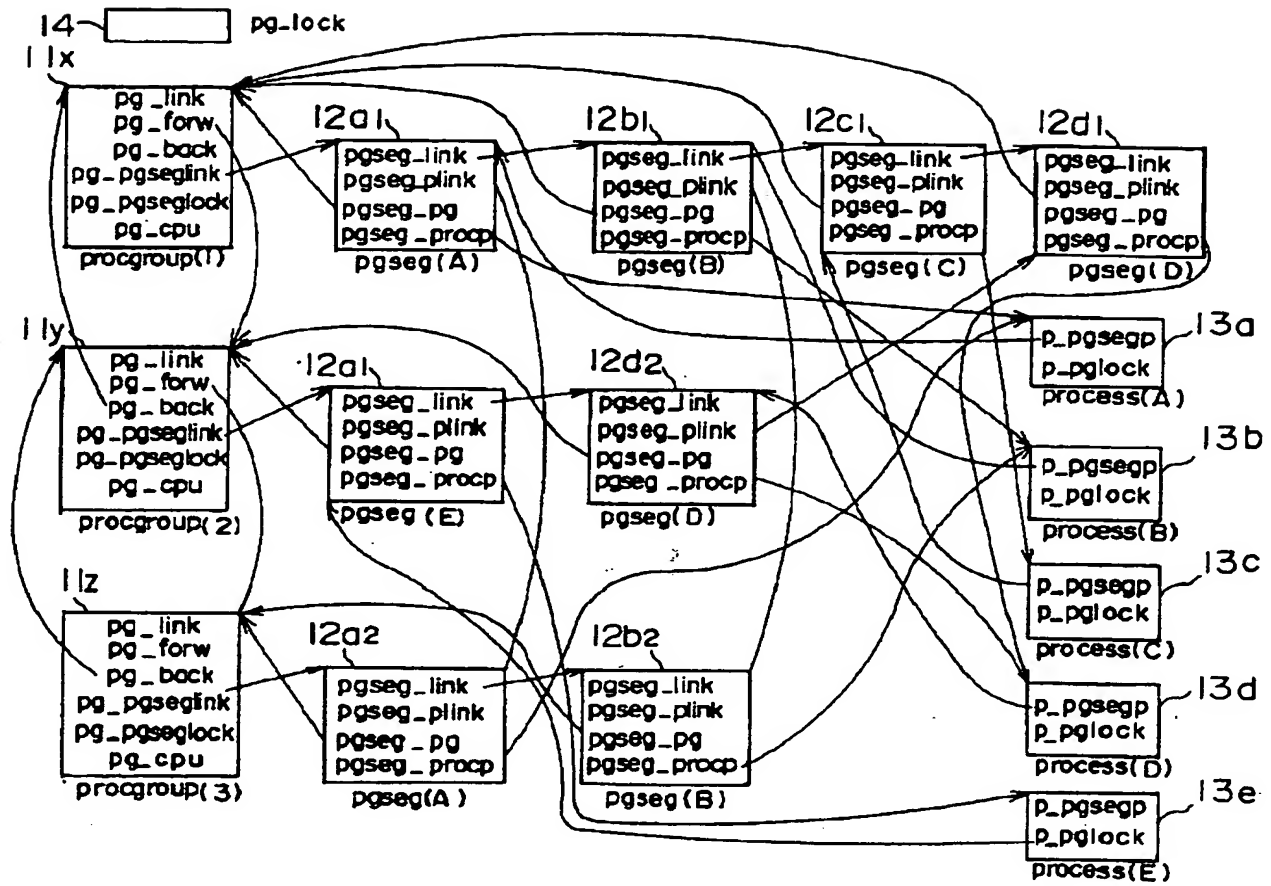
【図 6】



【図 8】



【図 9】



【図10】

